Design Assignment

on

Frequency Generation

Birla Institute of Technology and Science, Pilani

25th April 2019

Submitted By:

Group 11:

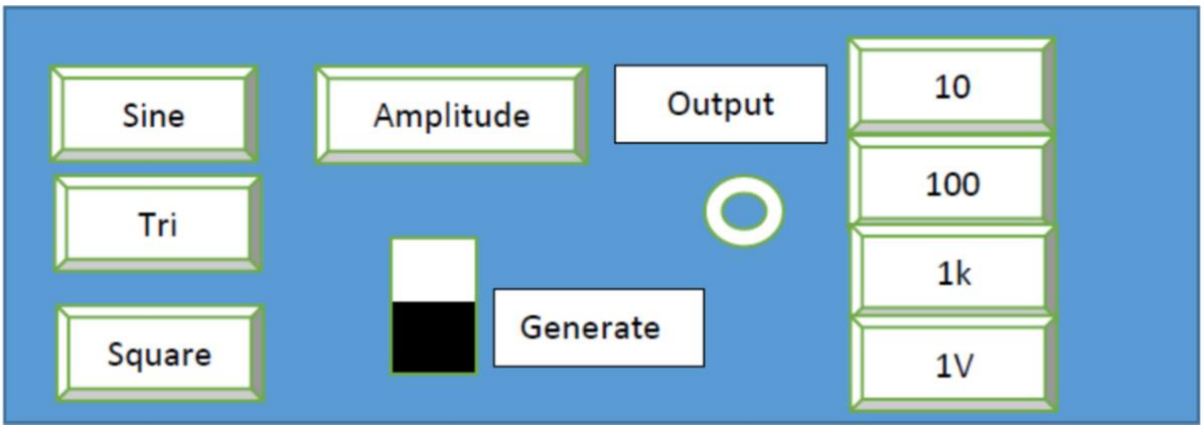
Purva Jain
Rishabh Jain
Vishesh Jain
Ananya Malhotra

Problem Statement

System to be Designed: Frequency Generation

Description: This system is used to generate a Sine/Triangular/Square waveform of Frequencies ranging from 10 Hz to 99KHz. Voltage is between 0-10V.

User Interface:



On system power up the user has to configure the desired type of waveform (square/triangle/square), frequency and amplitude. To generate a Square Waveform of Frequency 9.35 KHz the user has to press square key, followed by 1K Key- 9 Times, 1K Key – 4 Times, 100 Key –3 Times 10 Key- 5 Times. To select the Amplitude the user will have to press Amplitude key and then press the 1V key “n” number of times where “n” is the peak to peak amplitude of the waveform to be generated. (only integer values of output voltages needs to be generated) When generate switch should be turned on and then the frequency generation is enabled ie, the square waveform of that frequency will be generated. When frequency generation is enabled, if the user wants to change the waveform into another type for e.g. sine he just has to press sine. When a signal of different type/amplitude /frequency has to be generated, the user will have to turn-off the generate switch and then configure the function generator as mentioned above.

Design Specifications

1. The project aims to design a simulation of frequency generator circuit. The digital interface is given in the problem statement above.
2. Three types of waveforms can be generated by the above circuit:
    a. Sinusoidal
    b. Triangular
    c. Square
3. The keypad has buttons for selecting the type of waveform, the peak-to-peak amplitude of the waveform and the frequency.
4. The type of the waveform can be selected using the buttons for the type, namely SIN, TRI and SQU.
5. The peak to peak voltage can be selected by pressing the ‘1V’ button those many number of times. Only integer values of the peak to peak voltages can be generated by this device.
6. The frequency can be selected by the combination of three buttons, namely ‘1K’, ‘100’ and ‘10’. Example, to generate 9.25 KHz, ‘1K’ = 9 times, ‘100’ = 2 times and ‘10’ = 5 times.
7. After pressing the ‘Generate Button’, the waveform will be visible on the display.
8. After that, if we want to change the type of the waveform, that can be done by pressing the desired button then and there itself.
9. To generate a waveform with a different amplitude, one will have to switch off the ‘Generate’ button and start the procedure once again.

ASSUMPTIONS

The following assumptions were made to develop the program for the system:

- The user does not press a key corresponding to a frequency more than 9 time.
- Amplitude is entered by directly pressing the ‘1V’ key, those many number of times.
- Once the signal of specific frequency is generated, the user cannot change the frequency or the amplitude or the waveform. The system must be reset to change them.
- The user does not press more than one key at once.
- User cannot choose to select two different waveforms before pressing the generate key. If user wants to change the waveform, the user can do so while the signal is being produced, by pressing the relevant key.
- The triangle and sine waves are generated by using lookup tables and are therefore approximate.
- At the location FFFF0H, where the instruction pointer points on RESET of microprocessor, there exists a JUMP statement leading to the start of the code.

List of Components

- 8086 (MICROPROCESSOR)
- 2 X 6116 (RAM – 2K)
- 2 X 2732 (ROM - 4K)
- 2 X 74LS245 (OCTAL BUFFER)
- 3 X 74LS373 (OCTAL LATCH)
- 74LS04 (NOT GATE)
- 7432 (OR GATE)
- 74LS138 (3:8 DECODER)
- DIGITAL OSCILLOSCOPE
- 8253 (PROGRAMMABLE INTERVAL TIMER)
- 8284 (CLOCK GENERATOR)
- 8255 (PROGRAMMABLE PERIPHERAL INTERFACE)
- LM741 (OPAMP)
- DACO830 (DAC)
- 3X3 KEYPAD

I/O Map for 8255

Base Address: 00H

It is I/O mapped I/O System

The addresses of the ports are as follows:

PORT	ADDRESS
Port A	00H
Port B	02H
Port C	04H
Control Register	06H

Data lines: D0-D7 data lines of the microprocessor (as it is connected in even bank)

Control Word:

Group A: Mode 0 Group B: Mode 0

Port A: Input

Port B: Output

Port C upper: Output

Port C lower: Input

Hence, the control word is 10001010b

Which is written to the control register.

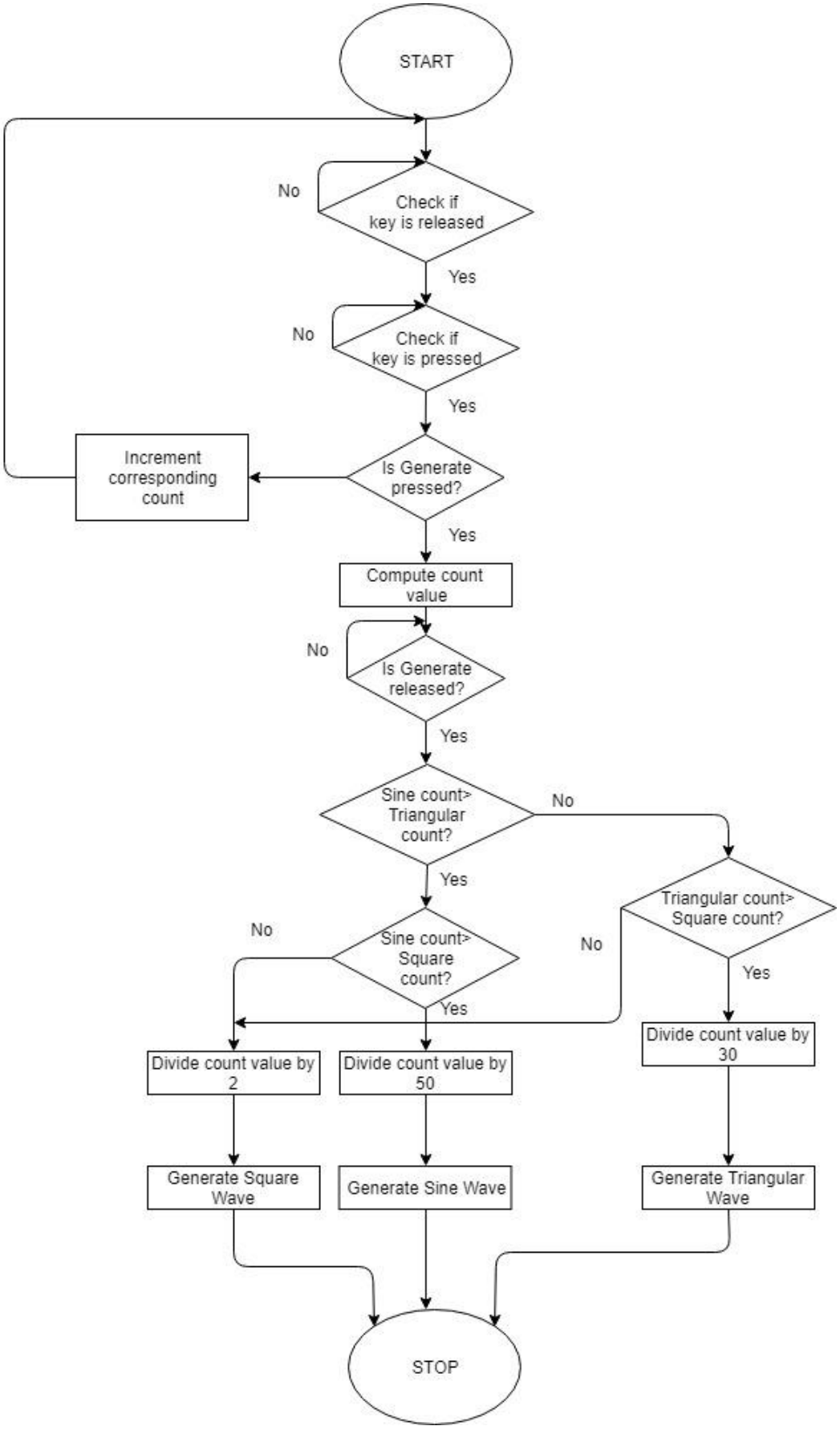
Address Map

Hex	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
04	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
06	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Memory Map

- RAM 1 (6116): 0200H – 03FFH
- RAM 2 (6116): 1000H – 17FFH
- RAM 3 (2732): 1800H – 27FFH
- RAM 4 (2732): 2800H – 37FFH

Flowchart


START
No
Check if key is released
Yes
No
Check if key is pressed
Yes
Increment corresponding count
Is Generate pressed?
Yes
Compute count value
No
Is Generate released?
Yes
Sine count> Triangular count?
No
Yes
Triangular count> Square count?
No
Sine count> Square count?
No
Yes
Yes
Divide count value by 2
Divide count value by 50
Divide count value by 30
Generate Square Wave
Generate Sine Wave
Generate Triangular Wave
STOP

Code:

```
#make_bin#

#load_segment=FFFFH#
#load_offset=0000H#

; initializing all the registers

#cs=0000H#
#ip=0000H#
#ds=0000H#
#es=0000H#
#ss=0000H#
#sp=FFFEH#
#ax=0000H#
#bx=0000H#
#cx=0000H#
#dx=0000H#
#si=0000H#
#di=0000H#
#bp=0000H#

; starting of the program

jmp st:
db 2042 dup(0)
st:  cli

; initializing all the variables to zero
```

```
one_k db 0
vfac db 0
sine_w db 0
triangular_w db 0
stepsize db 0
square_w db 0
one_hundred db 0
ten db 0
count dw 0
list db 13 dup(0)

; Giving names for the internal addresses of different ports of 8255

portA equu 00H
portB equu 02H
portC equ 04H
cregPPI equ 06H

; Giving names for the internal addresses of 8253

timer0 equ 08H
timer1 equ 0AH
timer2 equ 0CH
cregPIT equ 0EH

; Giving names to the different buttons hexcodes present on keypad..

SINbutton equ 66H
TRIbutton equ 56H
SQUbutton equ 36H
```

```
vbutton equ 65H
OKbutton equ 55H
HUNbutton equ 35H
TENbutton equ 33H
GENbutton equ 63H

; Initializing the segments to start of ram

mov     ax, 0200H
mov     ds, ax
mov     es, ax
mov     ss, ax
mov     sp, 0FFFEH
mov     ax, 00H
mov     vfac, al
mov     one_k, al
mov     vfac, al
mov     one_hundred, al
mov     ten, al
mov     sine_w, al
mov     triangular_w, al
mov     square_w, al

; Table to generate sine wave

lea     di,  list
mov     [di],128
mov     [di+1],144
mov     [di+2],160
mov     [di+3],176
```

```
mov     [di+4],191
mov     [di+5],205
mov     [di+6],218
mov     [di+7],228
mov     [di+8],238
mov     [di+9],245
mov     [di+10],251
mov     [di+11],254
mov     [di+12],255
mov       [di+13],254
mov       [di+14],251
mov       [di+15],245
mov       [di+16],238
mov       [di+17],228
mov       [di+18],218
mov       [di+19],205
mov       [di+20],191
mov       [di+21],176
mov       [di+22],160
mov       [di+23],144
mov       [di+24],128
mov       [di+25],127
mov       [di+26],111
mov       [di+27],95
mov       [di+28],79
mov       [di+29],64
mov       [di+30],50
mov       [di+31],37
mov       [di+32],27
mov       [di+33],17
```

```
mov       [di+34],10
mov       [di+35],4
mov       [di+36],1
mov       [di+37],0
mov       [di+38],1
mov       [di+39],4
mov       [di+40],10
mov       [di+41],17
mov       [di+42],27
mov       [di+43],37
mov       [di+44],50
mov       [di+45],64
mov       [di+46],79
mov       [di+47],95
mov       [di+48],111
mov       [di+49],127

; Initializing 8255 (setting it to i/o mode)
; Initializing the control register of 8255 accordingly

mov     al, 10001010b
out     cregPPI, al

; Keypad interfacing

key1:
mov       al, 00H
out     portC, al

; Checking for key release
```

```
key2:
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key2

mov      al, 00H
out         portC, al

; Checking for key press

key3:
in        al, portC
and     al, 70H
cmp     al, 70H
je      key3

; Once key press is detected, then find which row is the pressed key in

; if key is pressed in first column
mov     al, 06H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key4

; if key is pressed in second column
```

```
mov     al, 05H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key4

; if key is pressed in third column
mov     al, 03H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
je      key3

; Code reaches here once a key has been pressed and its hex code is stored in the al and bl registers
; Now we check which button that hexcode corresponds to:
; we will do OR of al and bl to store the hexcode corresponding to a particular button pressed

key4:or     al, bl
cmp     al, SINbutton
; If SIN button is pressed, then:
jnz     trib
inc     sine_w                  ;inc makes sine_w 1 which means it is selected
jmp     key1

trib:cmp     al, TRIbutton
```

```
; Else if TRI button is pressed, then:
jnz      squb
inc  triangular_w
jmp     key1

squb:cmp     al, SQUbutton
; Else if SQU button is pressed, then:
jnz     vfb
inc     square_w
jmp     key1

vfb:    cmp al, vbutton
;else if vbutton is pressed
jnz okb
inc vfac
jmp key1


okb:cmp     al, OKbutton
; Else, if 1K button is pressed, then:
jnz      hunb
inc     one_k
jmp     key1

hunb:cmp     al, HUNbutton
; Else, if 100 button is pressed, then:
jnz      tenb
inc     one_hundred
jmp     key1
```

```
tenb:cmp     al, TENbutton
; Else, if 10 button is pressed, then:
jnz      genb
inc     ten
jmp     key1

genb:cmp     al, GENbutton
; Else, if GEN button was pressed:
jz      end_k
jmp key1

end_k:

; Code reaches this point if GEN button is pressed.
; In that case, compute the count required to load in 8253 (PIT)

call computeCount

; BX register now stores the frequency in decaHertz

mov     dx, 00H
mov     ax, 10000
div     bx ; dividing 10000 by bx. Quotient stored in ax

i:  mov count, ax

; Calculated frequency present in count variable
; Storing count

mov     al, 00H
```

```
out     portC, al

; Wait for GEN key to be released

call waitForGEN

; BX now stores the value of (actual count * sampling rate)
; Here we have used the sampling rate of ((13*2)-1)*2 = 50

; now we will be selecting the wave form whose button is pressed maximum number of times
; If all have been pressed the same number of times, then sine wave will be selected

mov     al, sine_w
cmp     al, triangular_w
jl      slt
cmp       al, square_w
jg        sine_gen
jmp     sq_gen
slt:mov     al, triangular_w
cmp     al, square_w
jg        tri_gen
jmp     sq_gen

; Procedure to generate sine wave

sine_gen:
mov dx, portA
;mov dx, 00H
mov ax,count
mov bl,50
```

```
div bl
mov ah,00
mov bl, al

; Initializing timer
call initTimer
lea     si, list
mov     cl, 50
x99:
mov al, [si]
mul vfac
mov bl,10
div bl
mov [si],al
inc si
loop x99                        ;loop again and again to change values of sine table
according to given input..



l5:
lea     si, list
mov     cl, 50
l1:
mov     al, [si]
out     portA, al
call wait
J1: add     si, 01H
loop    l1
jmp     l5

; Code to generate triangular wave
```

```
tri_gen:
mov     dx, 00H
mov     ax, count
mov     bx, 30
div     bx
qr1:
mov     ah, 00
mov     bx, ax
; Initializing timer
call initTimer
mov al,25
mul vfac
mov vfac,al
mov ah,00h
mov bl,15
div bl
mov stepsize,al                    ;stepsize such that it takes 15 steps to reach max amplitude
mov bl,15
mul bl
mov vfac,al                ;vfac now is having the max amplitude

mov     al, 00H
g1:
out     portA, al
mov     bl, al
call    wait
mov     al, bl
add     al, stepsize
cmp     al, vfac
```

```
jnz     g1
g2:
out     portA, al
mov     bl, al
call    wait
mov     al, bl
sub     al, stepsize
cmp     al, 00H
jnz     g2
jmp     g1

; Code to generate square wave:

sq_gen:
mov dx, portA
mov ax, count
mov bx, 02H
div bx
mov bx, ax
mov al,25
mul vfac
; to set the amplitude of wave
mov vfac,al
mov ax,bx

; Initialize timer
call initTimer
mov     al, 80H
out     portA, al
s:  mov     al, 00H
```

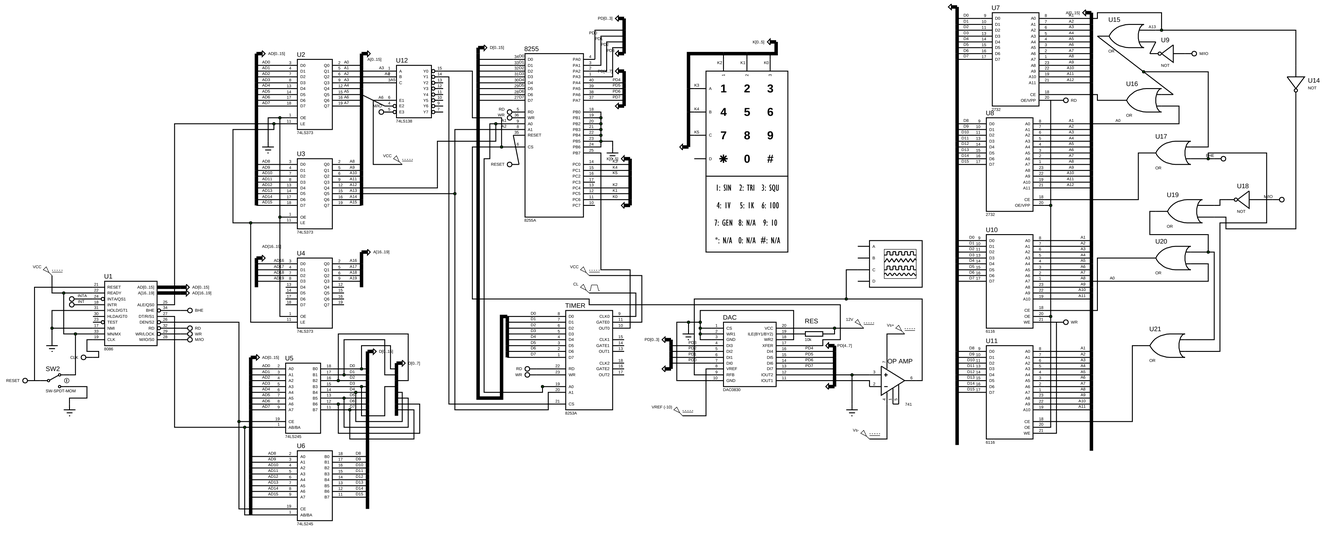
```
out     portA, al
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key
call    wait
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key


mov     al, vfac
out     portA, al
mov     al, vfac
out     portA, al
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key
call    wait
in      al, portC
and     al, 70H
cmp     al, 70H
jne     key
mov     al, vfac
out     portA, al
jmp     s

; Checking if a key is pressed
```

```
key:
mov     al, 06H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
jnz     k3

mov     al, 05H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
jnz     k3

mov     al, 03H
mov     bl, al
out     portC, al
in      al, portC
and     al, 70H
cmp     al, 70H
je      key

; If a key is pressed, we find out which one is pressed:

k3: or      al, bl
cmp     al, SINbutton
```

```
; If SIN button is pressed, then:
jz      sine_gen
cmp     al, TRIbutton
; Else, if TRI button is pressed, then:
jz      tri_gen
cmp     al, SQUbutton
; Else, if SQU button is pressed, then:
jz      sq_gen
; Else (i.e. if none of the waveform buttons were pressed), then:
jmp key

; Procedure to compute the value of count

computeCount proc
mov     bx, 00H
mov     al, 100
mul     one_k
add     bx, ax
mov     al, 0AH
mul     one_hundred
add     bx, ax
mov     al, ten
mov     ah, 00H
add     bx, ax
ret
endp

; Waiting procedure

wait proc
```

```
v1: in      al, portB
cmp     al, 00H
jne     v1
v2: in      al, portB
cmp     al, 80H
jne     v2
ret
endp

; Procedure to initialize the 8253 (PIT)

initTimer proc
; Initializing the timer with control word
mov dx, 0019H

mov     al, 00110110b
out     cregPIT, al

; Loading LSB of count value
mov     al, bl
out     timer0, al
; Loading MSB of count value
mov     al, bh
out     timer0, al

ret
endp

; Procedure to wait for GEN key to be released
```

```
waitForGEN proc
k1: in      al, portC
and     al, 70H
cmp     al, 70H
jnz     k1
ret
endp
```



FILE NAME:	circuit.pdsprj
DESIGN TITLE:	circuit.pdsprj
PATH:	E:\BITS\Study Material\Year 3 - Sem 2\Mu\PRES_Group\Final Project\circuit
BY: @AUTHOR	REV: @REV
DATE:	25-04-2018
PAGE:	1 OF 1
TIME:	11:46:18